Melody Extraction
Project Proposal
DSGA-1003 Machine Learning

Justin Mao-Jones, Junbo Zhao, Rita Li
Center for Data Science
New York University
justinmaojones, j.zhao, ml4713@nyu.edu

1 Introduction

We propose to build and evaluate machine learning models for the task of extracting melody from digital music files. Over the past decade, melody extraction has been an active research area in the music information retrieval research community [1]. It has many applications, including query by humming (e.g. hum a song into your phone and an app tells you what song it is), cover song identification, genre classification, and mood classification, among others.

Generally speaking, melody is the predominant pitch in a piece of music that captures the essence of a song. As a motivating example, the melody is the tune one might hum when asked "what does the song sound like?" Unfortunately, there does not seem to be a precise definition of melody. As a machine learning task, we require some sort of precise definition, and for this project we adopt the definition described by [1]:

> Melody is the fundamental frequency [1] from musical content with a lead voice or instrument. Melody extraction is the estimation of this melody from a single source.

While this definition is still open to interpretation, it can be used by human experts to generate the melody of a piece of music. Note that we constrain the definition of melody extraction to a single source, meaning that the melody is only coming from a single lead voice or instrument. The motivation for this simplification is that it can make the task of melody extraction easier.

Melody extraction is a supervised learning task and requires a set of labeled data. For this project, we propose to use MedleyDB [2], a dataset of annotated, royalty-free multitrack recordings that was curated primarily to support research on melody extraction. In section 1.1, we discuss the dataset further.

Many approaches to melody extraction have been attempted [1], including pure signal processing [3], dynamic programming [4], support vector machines [5], and hidden markov models [6] [8]. In this project, we plan to utilize SVM, HMM, and deep neural network architectures. We discuss these approaches further in section 4.3.

[1] Any audio signal can be represented as a sum of a series of sinusoids. The fundamental frequency of a signal defined as the lowest frequency in the series. It can be derived through the Fourier Transform.

1.1 Data

MedleyDB [2] consists of 122 multitracks, including stereo quality mixed audio, melody annotations, and stems [2]. The multitracks include songs from a variety of genres, including Singer/Songwriter, Classical, Rock, World/Folk, Fusion, Jazz, Pop, Musical Theatre, and Rap.

The audio files are provided in WAV format (44.1 kHz, 16 bit). In other words, each audio file contains 44,100 digital audio samples per second. Each audio file is accompanied by a single source melody annotation, provided in csv format. A melody label is a number that represents the predominant frequency over a pre-defined window of time. There are 256 melody labels per second. Each melody label overlaps roughly 172 audio samples.

One of the benefits of MedleyDB is that it was carefully curated to provide a complete set of melody annotations combined with high-quality songs. Thus, we are operating under the assumption that we do not have missing data and that the melody annotations perfectly overlap with the audio samples. Unfortunately, it would not be practically feasible to check this assumption thoroughly, but we are performing simple checks such as checking the length of song against the length of its corresponding melody annotation. So far, no issues have stood out.

An open question for our project team is whether or not we have enough data. On the one hand, 122 songs does not sound large. On the other hand, 256 melody labels per second, at an average of 3 minutes per song, corresponds to over 5 million training samples. So the question is whether or not there is enough variety in this dataset to generalize well to other datasets or even between the training and test split. Fortunately, there are additional datasets that we can utilize, such as RWC [7]. Time willing, we will try to add these datasets to our project.

1.2 Data Pre-processing

MedleyDB comes with an accompanying API for working with its data files, and so very little data pre-processing will be required. The majority of data processing work will be in the form of feature generation and generating training, validation, and test splits.

2 Problem Definition

As a machine learning task, melody extraction has two components:

1. Voicing detection (i.e. classification of whether or not the melody is present),
2. Melody pitch tracking.

The first task derives from the fact that sometimes there is no melody. The tasks can be approached in separate algorithms or in the same algorithm. Melody pitch tracking seems to be the more difficult task of the two.

2.1 Melody Pitch Tracking

Melody pitch tracking is the task of identifying the predominant frequency during a *frame* (a small time interval). We are effectively constrained to the frame size used in the MedleyDB melody annotations, and thus our frames are roughly 4ms.

An important modeling question is whether to predict melody labels through regression or classification. Regression can seem to be a natural fit, because frequency lies on a continuous spectrum. However, human music tends to be composed on a discrete scale, i.e. musical notes. We believe that regression could be used to predict melodies, and predictions could be refined by ”rounding” the predictions to the nearest note.

If using classification, then we would need to model all possible notes. For example, we could use the 88 keys found on a standard piano as the domain of our labels. Typically, most songs may not

[2] In a recording session, there is a separate microphone for each instrument (or sets of instruments), and thus there are separate recordings. For example, the singer is recorded separately from the guitar. A stem is one of these separate recordings. When added together the mix will sound like a complete song.

even cover this entire spectrum, and so we could reduce the size of our classification to only those notes present in our data. Converting melody annotations to discrete notes is a simple exercise.

3 Evaluation Metrics For Model Performance

Given that melody extraction consists of two tasks, it is natural to evaluate each task separately and together, thus yielding three different evaluation metrics. Typically, researchers use accuracy to evaluate performance [3], and so we will do the same here by following the MIREX accuracy definitions:

1. Voicing detection accuracy - TPR, i.e. probability that a frame that is actually voiced is predicted to be voiced,
2. Raw pitch accuracy - probability of a correct pitch value (within 1/4 tone[4]) given that the frame is voiced,
3. Overall accuracy = $\frac{TPC+TN}{TO}$
   TP = total number of frames correctly predicted as voiced
   TPC = total number of TP frames in which pitch was also correctly predicted
   TN = total number of frames correctly predicted as unvoiced
   TO = total number of predictions

4 Approaches

4.1 Feature Extraction

Since Music Information Retrieval (MIR) is closely related to the Speech Recognition (SR) community, it naturally absorbs feature descriptors and machine learning techniques from the Speech community, which has a relatively larger amount of literature. We briefly studied the marriage between the communities and settled a few methods that might be appropriate.

One of the difficulties in both MIR and SR is that, in order to progress in model performance, task-specific handcrafted features are sometimes needed. We acknowledge this, but have decided that we will start with off-the-shell features such as Mel-Frequency Cepstral Coefficients (MFCC) [1], Short-time Fourier Transform (STFT) and multi-resolution FFT (MRFFT) [8].

- **STFT**. STFT is a widely used signal preprocessing technique. In general, the signal is chunked into frames where STFT is applied to each chunked frame, with a window length typically assigned as 50 and 100ms. Frames can overlap in a sliding window fashion.
- **MRFFT**. Resolution issues inherently arise with Fourier transform. MRFFT overcomes this by taking frequency spectrum out of multi-resolution windows.
- **MFCC**. This has been the dominant feature descriptor in Speech community over the past 30 years. It is basically a linear cosine encoding of the log power spectrum on a mel-scale of frequency which biologically originated from human's ears.
- **Dictionary Learning**. Dictionary learning is an adaptive content-based feature self-learning approach. Its goal basically is to get local descriptions by learning a linear combination of a pre-defined dictionary. The weights of the words in the dictionary are the new representation of the local window. The process of getting the dictionary is unsupervised; K-means and K-SVD [9] are two common methods to obtain this dictionary.

In general, Fourier transform intuitively seems to be natural to the problem of melody extraction. The human auditory system naturally perform a fourier analysis in the conversion of sound to neural impulses. In addition, the fourier transform generates a time series of frequency vectors, and the melody traverses across these vectors through time.

[3] http://www.music-ir.org/mirex/wiki/2014:Audio_Melody_Extraction

[4] a tone is a step between two keys on a piano; e.g. G is a tone higher than F.

4.2 Melody Specific Challenges

There are two major difficulties with melody. First, the melody is often mixed in with other musical signals. For example, the melody might be produced by the singer, but there is also a guitar, a bass guitar, and drum beats playing at the same time. We are not yet entirely sure how to tackle this challenge. It will be a topic that we continue to discuss and focus on throughout the project.

Second, melody is somewhat context dependent. For example, it might actually be impossible for any algorithm to detect discernable melody patterns in a 4 ms window. Intuitively, it seems that one must listen to the music surrounding that specific time frame in order to determine the melody. Thus, the features of a specific sample instance should include audio signal information from surrounding time frames. The number of such features could be potentially large, possibly huge. How large of a time window is needed to identify a melody? If the answer is one tenth of a second, then we would have 4410 audio samples per melody label. If this is converted to an MFCC vector, then we could have roughly 170,000 features per melody label. An additional related question is whether or not future information is required to predict melodies.

Looking at our data we have deduced that a window of approximately 12 ms is sufficient to capture at least two cycles of the fundamental frequency of the music. Thus, we initially begin with this size of a window in order to train baseline algorithms for proof of concept. Once that is complete, we can start increasing the size of the windows.

4.3 Modeling

We have elected to use hidden markov models (HMM) with gaussian mixture models (GMM), support vector machines (SVM), and deep learning as an initial starting point for modeling melody. We describe these approaches below.

- **SVM**. Using SVM for melody extraction has been applied before to build classifiers. One of the advantages of SVM is that it requires no prior knowledge of how they are presented in the examined features. We want to examine the problem by training classification for voice detection stage and regression for raw pitch estimation. In this project, we would love to explore linear and RBF kernels, since linear model will have lower computational costs, and more tractable and with l_1 and l_2 penalty options thus we can adjust the model accordingly based on the features' sparsity. However, RBF kernel is the most popular kernel being used when train a SVM model and it will fit the model more flexibly, but at higher computational cost. It depends on the feature dimensions from the data since if it is low dimension which might not be linearly separable in this case. We want to optimize the problem first in primal space. If it is doable, we will proceed to dual space.

- **HMM-GMM**. HMM seems like a natural fit for melody extraction: given a previous state (or set of previous states) and some observed features that describe the audio signal around the current melody frame, what is the most likely melody state? HMM has been extremely successful in SR and has been successful in melody extraction [6]. Using HMM, our hidden states would be musical notes corresponding to melody frequencies. As for the observation states, in traditional SR, gaussian mixture models are often used to encode MFCC output into a set of observation states, and each frame is tagged to exactly one of these states (the one corresponding to the highest posterior probability). We will use this framework as an initial starting point, because it is relatively straightforward. Down the road we may try to incorporate CNN output (see below) into the observation states and other methods for defining the hidden state-observation transition matrix.

Inspired by the recent success of deep learning in SR, an alternative approach is to apply end-to-end learning techniques that derive useful information from the raw audio waves [J4]. Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) are two predominant choices modeling sequential data. Both of them can be applied to the raw audio input or Fourier transformed features and trained by gradient based method using back propagation. Thus, we also plan to experiment with CNN. Time-willing, we will also experiment with RNN. We expect that either the CNN or RNN will produce the best results.

- **CNN**. CNN operates between Temporal Convolution, Temporal Pooling and Non-linear activations. On top of the network framework, a classifier or regressor would be placed to finish the job of classifying or regressing.
- **RNN**. Differing from feedforward neural network, RNN uses the internal memory to process signals in such a way that RNN output from previous time steps are used as inputs for subsequent time steps.

We will also look for ways to incorporate CNN and/or RNN features into SVM and HMM-GMM models.

4.4 Baseline Models

It is important to have a baseline performance to measure the performance of a model against. We will use the following baseline models:

- Regression:
  Pitch Tracking: Predict the average melody pitch
- Classification:
  Voicing Detection: Predict the majority class (on/off)
  Pitch Tracking: Predict the majority class

4.5 Data Augmentation

In recent years, computer vision has seen a huge advancement in the predictive power of machine learning algorithms, primarily via deep learning approaches. As part of that advancement, researchers have found that data augmentations can provide an added boost to algorithm performance.

The idea behind data augmentation can be explained using the following example. The essence of a song, and corresponding melody, is the same regardless of what volume it is. A machine learning algorithm trained on multiple versions of the same song (i.e. at different volumes) will likely show better performance than an algorithm trained on only one version.

Thus, we plan to do the following data augmentations:

- Adjust volume of songs
- Add/remove non-melodic stems from songs
- Add white noise to songs

This list may grow as we discover more data augmentations.

4.6 Training/Validation/Testing

We will be splitting our training, validation, and testing sets on songs. In other words, no songs will overlap between sets. This intuitively makes sense. In a real world application, a melody extraction procedure would be applied to songs it had never seen before.

Given the limited number of songs in our dataset, we may use cross-validation instead of separate training and validation sets. Initially, we will simply use 25 songs for the test set, 25 songs for validation, and the remaining 72 songs for training.

5 Anticipated Timeline

- April 3 - Generate baseline models, feature generation complete, begin building and running models models
- April 12 - Successful runs of SVM, HMM, and CNN complete, begin hyperparameter tuning and grid search

- April 19 - Continue hyperparameter tuning and grid search.
- April 26 - SVM, HMM, CNN complete, begin analyzing results
- May 3 - Analysis of model results complete. Using analysis, determine new modeling directions. Begin writing report.
- May 10 - 2nd round of models complete and finalize report
- May 14 - Final report complete.

References

[1] J. Salamon, E. Gomez, D. P. W. Ellis and G. Richard, "Melody Extraction from Polyphonic Music Signals: Approaches, Applications and Challenges", IEEE Signal Processing Magazine, 31(2):118-134, Mar. 2014.

[2] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam and J. P. Bello, "MedleyDB: A Multitrack Dataset for Annotation-Intensive MIR Research", in 15th International Society for Music Information Retrieval Conference, Taipei, Taiwan, Oct. 2014.

[3] J. Salamon and E. Gmez, "Melody Extraction from Polyphonic Music Signals using Pitch Contour Characteristics", IEEE Transactions on Audio, Speech and Language Processing, 20(6):1759-1770, Aug. 2012.

[4] V. Rao and P. Rao, Vocal melody extraction in the presence of pitched accompaniment in polyphonic music, IEEE Trans. Audio, Speech, Lang. Processing, vol. 18, no. 8, pp. 21452154, Nov. 2010.

[5] G. Poliner and D. Ellis, A classification approach to melody transcription, in Proc. 6th Int. Conf. Music Information Retrieval, London, Sept. 2005, pp. 161166.

[6] M. Ryynnen and A. Klapuri, Automatic transcription of melody, bass line, and chords in polyphonic music, Comput. Music J., vol. 32, no. 3, pp. 7286, 2008.

[7] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka: RWC Music Database: Popular, Classical, and Jazz Music Databases, Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR 2002), pp.287-288, October 2002.

[8] T.-C. Yeh, M.-J. Wu, J.-S. Jang, W.-L. Chang, and I.-B. Liao, A hybrid approach to singing pitch extraction based on trend estimation and hidden Markov models, in IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP), Kyoto, Japan, Mar. 2012, pp. 457460.

[9] Aharon, Michal and Elad, Michael and Bruckstein, Alfred, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," in IEEE Transactions on Signal Procesing 2006, Vol 54 No 11 pp. 4311.